Parametric t-stochastic neighbor embedding with quantum neural network

Yoshiaki Kawase,[1,*] Kosuke Mitarai,[1,2,3,†] and Keisuke Fujii[1,2,4,‡]

[1]*Graduate School of Engineering Science, Osaka University 1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan*

[2]*Center for Quantum Information and Quantum Biology, Osaka University, Osaka 560-8531, Japan*

[3]*Japan Science and Technology Agency (JST), Precursory Research for Embryonic Science and Technology (PRESTO), 4-1-8 Honcho, Kawaguchi, Saitama 332-0012, Japan*

[4]*RIKEN Center for Quantum Computing, Wako Saitama 351-0198, Japan*

(Received 3 March 2022; accepted 22 November 2022; published 19 December 2022)

t-stochastic neighbor embedding (t-SNE) is a nonparametric data visualization method in classical machine learning. It maps the data from the high-dimensional space into a low-dimensional space, especially a two-dimensional plane, while maintaining the relationship or similarities between the surrounding points. In t-SNE, the initial position of the low-dimensional data is randomly determined, and the visualization is achieved by moving the low-dimensional data to minimize a cost function. Its variant called *parametric t-SNE* uses neural networks for this mapping. In this paper, we propose to use quantum neural networks for parametric t-SNE to reflect the characteristics of high-dimensional quantum data on low-dimensional data. We use fidelity-based metrics instead of Euclidean distance in calculating high-dimensional data similarity. To verify our proposed method, we visualize both classical (Iris dataset) and quantum (time-depending Hamiltonian dynamics) data for classification tasks. Since this method allows us to represent a quantum dataset in a higher dimensional Hilbert space by a quantum dataset in a lower dimension while keeping their similarity, the proposed method can also be used to compress quantum data for further quantum machine learning.

DOI: 10.1103/PhysRevResearch.4.043199

I. INTRODUCTION

Visualization of high-dimensional data is an important subfield of machine learning. It allows us to intuitively interpret the data and understand possible patterns in them. As visualization often involves mapping of the original data to a low, typically two- or three-dimensional space, the techniques for visualizations are also useful for compression of data or preprocessing before applying other machine learning techniques. Prototypical examples of visualization techniques include t-stochastic neighbor embedding (t-SNE) [1].

Such techniques have been proven to also be useful for machine learning of quantum states. The authors of Ref. [2] applied various visualization methods to detect quantum phase transitions in the Hubbard model, where they generated states by quantum Monte Carlo simulations. They found that t-SNE is the most promising technique in this type of the application. The authors of Ref. [3] applied t-SNE to visualize quantum states represented as matrix product states. They successfully visualized quantum phase transitions in spin models such as the transverse field Ising model.

*yoshiaki.kawase@qc.ee.es.osaka-u.ac.jp

†mitarai@qc.ee.es.osaka-u.ac.jp

‡fujii@qc.ee.es.osaka-u.ac.jp

However, the application of machine learning techniques running on classical computers for this purpose is intrinsically limited to the case where the target quantum states have efficient classical representations. The use of quantum computers can provide an advantage in widening the range of quantum states that can be used as inputs. In fact, in Ref. [4], identification of the topological quantum phase has been proposed by using a clustering algorithm using a quantum computer. This approach is motivated by the fact that there is a family of quantum states that are useful for machine learning of classical data but cannot be efficiently represented by classical computers [5].

Here, we propose to use quantum neural networks (QNNs), which use a parametric quantum circuit to construct a machine learning model, for visualization of quantum states. Our proposal is based on parametric t-SNE [6], which is a visualization technique where we employ neural networks to map high-dimensional data into low-dimensional space. The mapping is optimized to keep the similarity of the data points, which is defined from the distance in the respective spaces, unchanged. Our idea is to use QNNs instead of the classical neural network. This allows us to directly use quantum states as inputs, which may be useful for studying complex quantum systems and certain machine learning problems that are hard classically. The similarity of the quantum states can be defined from fidelity(like) measures. Our method optimizes the parameters in a QNN so that the respective quantum states are mapped to low-dimensional points, which are defined as expectation values of certain observables at the output of the QNN, while maintaining the similarity among the points.

We also conduct numerical verification of our proposed method. First, we use the Iris flower dataset [7] to test if it can be successfully applied to classical data. Second, we visualize the quantum states time-evolved under the transverse-field Ising Hamiltonian. In the visualization of quantum data, we could not effectively generate a visualization with distinct clusters when using the default cost function usually utilized for parametric t-SNE. To deal with this problem, we introduce a hyperparameter into the cost function to adapt the scaling of the low-dimensional data.

With these successful demonstrations, we believe that the proposed method would be a powerful tool to visualize and analyze both classical and quantum datasets.

This paper is organized as follows. In Sec. II, we briefly review the t-SNE. In Sec. III, we describe our proposed method. In Sec. IV, we perform numerical experiments to verify our proposed method. In Sec. V, we describe the conclusions and future work.

II. BACKGROUND

A. t-SNE

The t-SNE [1], also called nonparametric t-SNE, is a classical machine learning method for visualizing high-dimensional data. The idea of t-SNE is to map data points in the original high-dimensional space to points in a low-dimensional space while keeping the similarity among the points. The map is determined by minimizing the Kullback-Leibler (KL) divergence between the similarity of data distributions in the high- and low-dimensional space.

In detail, the t-SNE defines the similarity between a high-dimensional data point x_i and another data point x_j by the following joint probability [1]:

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N}, \tag{1}$$

where

$$p_{j|i} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k\neq i}\exp\left(-\frac{\|x_i - x_k\|^2}{2\sigma_i^2}\right)}, \qquad p_{ii} = 0, \tag{2}$$

and σ_i are parameters determined from the following quantity called perplexity of data x_i:

$$\mathrm{Perp}_i = 2^{-\sum_j p_{j|i}\log_2 p_{j|i}}.$$

The value of σ_i is set to make Perp_i a user-specified value (typically between 5 and 50). The similarity between the low-dimensional data points y_i and y_j is defined by the following equation using Student t distribution with one degree of freedom [1]:

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_k \sum_{l\neq k}(1 + \|y_k - y_l\|^2)^{-1}}, \qquad q_{ii} = 0. \tag{3}$$

The t-SNE determines a low-dimensional point y_i corresponding to a data point x_i by iteratively minimizing the cost function $C(\{y_i\})$ defined as the KL divergence between a joint probability distribution in the high- and low-dimensional data:

$$C(\{y_i\}) = \sum_i \sum_j p_{ij}\log\frac{p_{ij}}{q_{ij}}. \tag{4}$$

The gradient of the cost function for y_i is described as

$$\frac{\partial C}{\partial y_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}. \tag{5}$$

In optimizing the cost function, each y_i is initially placed in a random position and moved to minimize the cost function.

B. Parametric t-SNE

Parametric t-SNE [6] is a variant of t-SNE which discards the direct optimization of the low-dimensional points $\{y_i\}$ but uses a neural network to map x_i to y_i. In this method, each low-dimensional point y_i is generated by $y_i = f(x_i|\theta)$, where $f(x_i|\theta)$ is an output of the neural network with an input x_i and network weight θ. We optimize θ to minimize the cost function in Eq. (4). An important distinction between the t-SNE and the parametric t-SNE is that the latter can easily generate a low-dimensional point for a new input as we explicitly construct the mapping from x_i to y_i.

C. Application of t-SNE for quantum systems

Recently, t-SNE has also been applied to the field of quantum physics by Yang *et al.* [3]. They considered the visualization of quantum phase transitions by applying t-SNE to (approximate) ground states $|\psi_i\rangle$ of certain Hamiltonians H_i. Their approach is to use

$$p_{j|i} = \frac{\exp\left[-\frac{d(\psi_i,\psi_j)^2}{2\sigma_i^2}\right]}{\sum_{k\neq i}\exp\left[-\frac{d(\psi_i,\psi_k)^2}{2\sigma_i^2}\right]}, \tag{6}$$

instead of Eq. (2), where the distance $d(\psi_i, \psi_j)$ is defined by the negative logarithmic fidelity:

$$d(\psi_i, \psi_j) = -\log(|\langle\psi_i|\psi_j\rangle|). \tag{7}$$

It has been shown that this approach can successfully visualize and identify quantum phase transitions of one-dimensional spin chains. In Ref. [3], the authors also considered the visualization of classical data by using the so-called *quantum feature map* via t-SNE. We share their idea in the sense of visualization of quantum states via t-SNE. However, our approach differs essentially from Ref. [3] in that a parameterized quantum model is employed to yield low-dimensional data of parametric t-SNE.

III. PARAMETRIC t-SNE WITH QUANTUM CIRCUITS

In this paper, we propose using quantum circuits to construct the parametric model $f(x|\theta)$ to generate low-dimensional data. The procedure of our proposed method is shown in Fig. 1. More concretely, using a parameterized unitary circuit $U(x,\theta)$ that depends on both a d-dimensional input x and trainable parameters θ, we generate a d'-dimensional data point y by expectation values of d' observables $\{O_\mu\}_{i=1}^{d'}$. Then we minimize the cost function

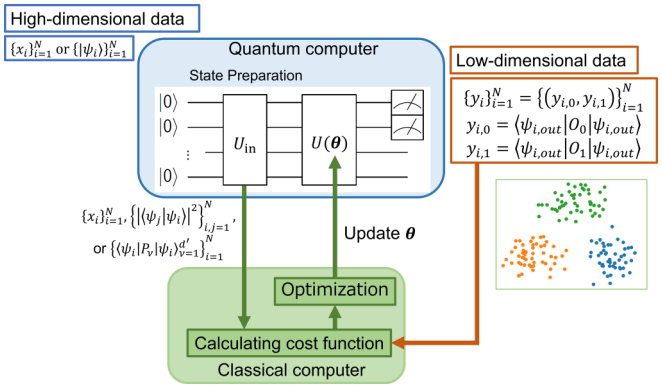

FIG. 1. Procedure of visualizing classical or quantum data. Specifically, we visualize classical and quantum data by mapping the high-dimensional data to the low-dimensional one. We prepare a quantum state by a parametrized quantum circuit for classical data or by time-evolution under a Hamiltonian for quantum data. The mapping is performed by a parametrized quantum circuit trained by minimizing a cost function to maintain the similarity of surrounding points. The similarity of high-dimensional data is defined by either expectation values, fidelities, or classical data. The similarity of low-dimensional data is determined by expectation values, and we plot them in a two-dimensional plane.

defined in Eq. (4) by optimizing the parameters θ. The concrete algorithm for classical inputs is as follows:

(1) Compute p_{ij} for all pairs from $\{x_i\}_{i=1}^N$.

(2) For all $\{x_i\}_{i=1}^N$ and $\{O_\mu\}_{\mu=1}^{d'}$, evaluate $y_\mu(x_i,\theta) = \langle 0|\, U^\dagger(x_i,\theta) O_\mu U(x_i,\theta)\,|0\rangle$ using a quantum computer.

(3) Compute q_{ij} for all pairs from $y_\mu(x_i,\theta)$ on a classical computer.

(4) Compute $C[\{y_\mu(x_i,\theta)\}_{i=1}^N]$ on a classical computer.

(5) Update θ so that C is minimized.

After the convergence, we expect the above protocol to find a quantum circuit $U(x,\theta)$ which can map x_i to a low-dimensional space while preserving the similarity based on distances among the data.

Note that the choice of $U(x,\theta)$ and O_μ affects the outputs y_i. First, we explain how to choose observables. We choose local observables to alleviate a vanishing gradient problem, which makes optimization hard. This is because the gradient is less likely to disappear using local observables than global observables [8]. Moreover, if we suppose $U(x,\theta) = U(\theta)U_{\rm in}(x)$ as in Fig. 1, our quantum model means that quantum states representing data are measured on bases optimized by a parameterized quantum circuit $U(\theta)$. Let us define $|\psi_{{\rm in},i}\rangle = U_{\rm in}(x_i)|0\rangle$, the output values $y_\mu(x_i,\theta)$ are described by $\langle\psi_{{\rm in},i}|U(\theta)^\dagger O_\mu U(\theta)|\psi_{{\rm in},i}\rangle$. This means that the encoded data $|\psi_{{\rm in},i}\rangle$ is measured by the observables $U(\theta)^\dagger O_\mu U(\theta)$. If the parameterized quantum circuit is deep enough to transform the observables appropriately, the choice of observables O_μ is not likely to have much effect on results. In our numerical experiments, we choose local observables acting on a single qubit. Next, we comment on the design of a quantum circuit. In our numerical experiments, we use hardware efficient ansatz, which is widely used in the study of quantum machine learning. However, this choice is only for proof-of-concept numerical demonstration, and important future work should explore what types of quantum circuits are suited for our proposed algorithm. The desirable design of the quantum circuit for quantum machine learning is actively studied [9,10]. At least, we can say that linear data encoding and transformation are preferred if the data are linearly separable. Otherwise, we have to encode data with nonlinearity, e.g., using data re-uploading [10].

An interesting extension of the above protocol is to take a quantum dataset consisting of quantum states $\{|\psi_i\rangle\}_{i=1}^N$ as the input data.

Here, we can think of several ways to define the similarity of quantum data in a high-dimensional space. One possible way is to generate high-dimensional classical data by measuring a set of observables $\{P_\nu\}_{\nu=1}^d$. Then their expectation values:

$$\{\langle P_\nu\rangle\}_{\nu=1}^d, \tag{8}$$

are used as the high-dimensional data. Another possibility is to use a distance function of two quantum states defined via fidelity:

$$d(|\psi_i\rangle, |\psi_k\rangle) = \sqrt{1 - |\langle\psi_i|\psi_k\rangle|^2}. \tag{9}$$

Then the similarity of two quantum states is defined as follows:

$$p_{j|i} = \frac{\exp\left[-\frac{d(|\psi_i\rangle,|\psi_j\rangle)^2}{2\sigma_i^2}\right]}{\sum_{k\neq i}\exp\left[-\frac{d(|\psi_i\rangle,|\psi_k\rangle)^2}{2\sigma_i^2}\right]}. \tag{10}$$

This choice enables us to readily calculate $d(\psi_i,\psi_k)$ on a quantum computer using standard techniques for overlap measurement such as swap test [11].

On the other hand, for the low-dimensional space, we can use $U(\theta)$, which only depends on the trainable parameters θ at step (2) in the classical input case since we do not have classical input x. For the cost function C, we can use the same formulation as above, that is, we measure $y_\mu(\psi_i,\theta) = \langle\psi_i|\, U^\dagger(\theta) O_\mu U(\theta)\,|\psi_i\rangle$ for a certain set of observables $\{O_\mu\}$ and compute $C(\{y_\mu(\psi_i,\theta)\}_{i=1}^N)$.

Note that, if we employ a quantum feature map:

$$|\psi_i\rangle = U_{\rm in}(x_i)|0\rangle, \tag{11}$$

the similarity of classical input data $\{x_i\}$ in high-dimensional space can also be calculated via the fidelities of the quantum states.

The possible advantage of the proposed method solely depends on the fact that there are quantum circuits that are hard to simulate classically [12–14]. This means that we may be able to construct the map $f(x|\theta)$ which cannot be expressed by neural networks. In fact, it is proven that, when using a very specific dataset, there is a machine learning task with rigorous quantum advantage [5]. However, the usefulness of quantum circuits for modeling practical classical data is, in general, still in question. Since one of our approaches assumes that the high-dimensional data are quantum data, we expect that the quantum model has an advantage in reproducing its similarity in low dimensions. We leave this research direction as an important future topic to explore.

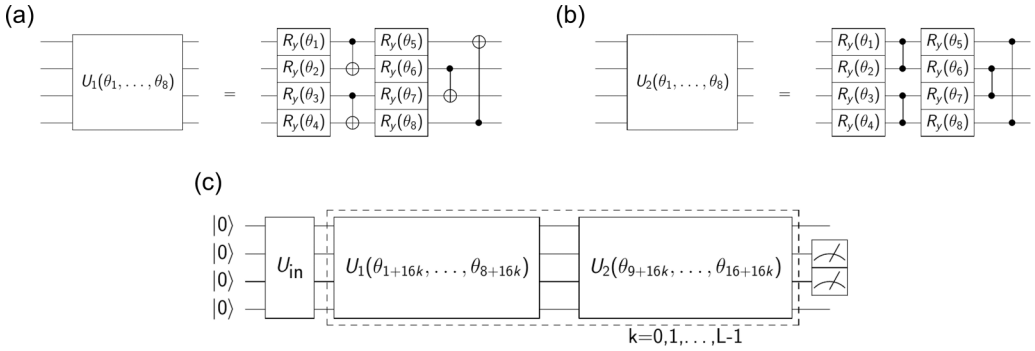

FIG. 2. Quantum circuit to visualize data: We input data by U_{in} and alternately act U_1 and U_2 on the quantum state. The quantum circuit (a) U_1 and (b) U_2.

IV. NUMERICAL EXPERIMENTS

Here, we perform the numerical simulations of the proposed method. First, let us describe the tools used in the experiments. We use QULACS [15] to simulate a quantum circuit. We make a python wrapper to work with PYTORCH [16] to use the loss function and optimizer. Our optimization is performed by Sharpness-Aware Minimization (SAM) [17,18] with ADAM [19] as the base optimizer. The overview of SAM is to consider the cost to which the l2-regularization term is added, find the gradient at the point where that cost is the highest in the neighborhood, and descend from the current point according to that gradient.

Note that, in this section, we denote X_i as the Pauli X acting on ith qubit, and also Z_i as the Pauli Z acting on ith qubit.

A. Visualizing classical data

1. Application of parametric t-SNE with quantum circuits

We visualize the Iris flower dataset [7] with our proposed method. The dataset contains three classes and consists of four features. We normalize each feature between -1 and $+1$. The construction of our ansatz $U(x,\theta)$ and observables $\{O_\mu\}$ for the low-dimensional space are as follows. Let us define $R_y(\theta) = \exp(i\theta\hat{Y}/2)$ and

$$U_{\text{in}}(x_i) = R_y(x_{i,1}) \otimes R_y(x_{i,2}) \otimes \cdots \otimes R_y(x_{i,d}),$$

where $x_{i,j}$ denotes the jth element of ith data x_i. Also, let us define U_1 and U_2 as shown in Fig. 2. Here, U_1 and U_2 are commonly used in hardware efficient ansatz. We use the combination of them as $U(x,\theta)$, defined by

$$U(x,\theta) = \prod_{k=0}^{L-1} [U_2(\theta_{9+16k:16+16k})U_1(\theta_{1+16k:8+16k})]U_{\text{in}}(x), \tag{12}$$

where $\theta_{i:j}$ denotes the $(j-i+1)$-dimensional vector containing ith to jth elements of θ. Specifically, we set the circuit depth $L=4$. As for the output, we set $d'=2$ and $\{O_\mu\} = \{X_2, X_3\}$ to visualize the Iris dataset on a two-dimensional plane. As we explained in Sec. III, we choose local observables to alleviate the vanishing gradient problem.

We perform the simulation under the above settings and plot the results in Fig. 3. This figure shows that the data are clustered for each type of Iris flower. Figure 3(a) on the left shows the visualization by the classical machine learning method of t-SNE and Fig. 3(b) in the middle by the proposed method with similarity based on Euclidean distance. In both two cases, the data are clustered by Iris flower varieties.

2. Visualization with infidelity distance measure

Next, we show that the proposed method can be correctly visualized by calculating the similarity of quantum states. To this end, we will use, as a test case, a set of quantum states generated by a quantum feature map from the Iris dataset before running the method on a truly physically meaningful quantum state. We perform the parametric t-SNE using quantum features $|\psi_i\rangle$ defined as

$$|\psi_i\rangle = U_{\text{in}}(x_i)\,|0\rangle\,, \tag{13}$$

and the cost function associated with the distance measure defined in Eq. (9). We show the result in Fig. 3(c), which implies that this protocol also works.

B. Visualization of quantum data

1. Visualization based on observables

In this section, we perform the visualization of quantum states. As an example, let us consider the time-dependent two-body transverse field Ising model, which is employed in quantum annealing [20]. We prepare quantum states time-evolved under the following Hamiltonian:

$$H = \left(1-\frac{t}{\tau}\right)\sum_i h_i X_i + \left(\frac{t}{\tau}\right)\sum_{i<j} J_{ij} Z_i Z_j, \tag{14}$$

where τ denotes the total simulation time of the Hamiltonian. We perform the simulation under the Trotter decomposition [21,22], defined by

$$\begin{aligned}\exp[-i(H_1+H_2)t] &= \lim_{m\to\infty}\left[\exp\left(-\frac{iH_1 t}{m}\right)\exp\left(-\frac{iH_2 t}{m}\right)\right]^m \\ &\simeq \left[\exp\left(-\frac{iH_1 t}{m}\right)\exp\left(-\frac{iH_2 t}{m}\right)\right]^m.\end{aligned}$$

We set the number of qubits as four, $\tau = 40$, time step $\Delta t := \tau/m = 0.01$, $h_i = -1$, and J_{ij} as a uniform random number between -1 and 0 or 0 and $+1$.

We prepare 200 random Hamiltonians. Half of them have all positive J_{ij}, corresponding to being antiferromagnetic

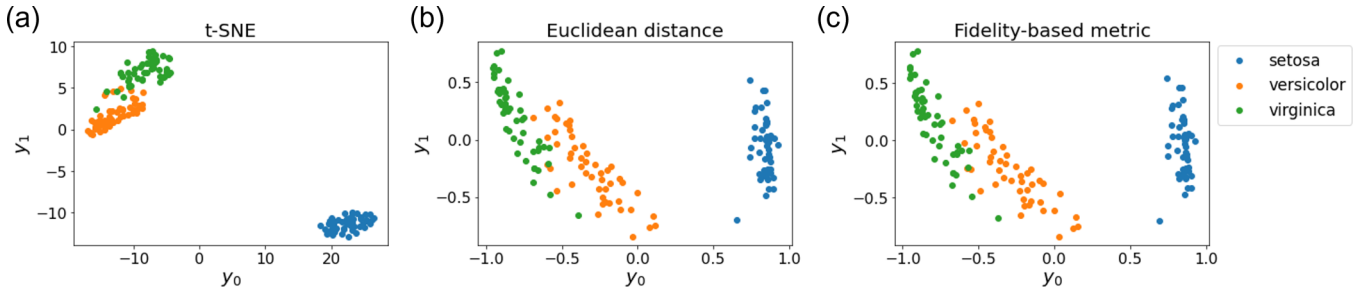

FIG. 3. Visualization of the Iris flower dataset (a) by the classical machine learning method of t-stochastic neighbor embedding (t-SNE), (b) by the proposed method with the similarity based on Euclidean distance between high-dimensional data, and (c) by the proposed method with the similarity based on the fidelity-based metric between high-dimensional data.

(AFM), and the rest have all negative J_{ij}, corresponding to being ferromagnetic (FM).

The dynamics is performed as follows. Since we prepare the ground state of $\sum_i h_i X_i$ as the initial state, the adiabatic evolution results in a ground state $(|0\rangle^{\otimes n} + |1\rangle^{\otimes n})/\sqrt{2}$ of $\sum_{i<j} J_{ij} Z_i Z_j$, which does not depend on the specific values of J_{ij}. On the other hand, AFM Hamiltonians have different ground states depending on the values of J_{ij}, and therefore, the visualization results in a scattered pattern.

Here, we visualize the quantum state at every 1000 Trotter steps ($t = 10, 20, 30$, and 40) by making the models for each step. As explained before, we consider the two methods to compute similarity of the quantum states in high-dimensional space. The first method is to consider the expectation values of input quantum states as high-dimensional data:

$$x_i = (\langle\psi_{i,\mathrm{in}}|X_1|\psi_{i,\mathrm{in}}\rangle, \langle\psi_{i,\mathrm{in}}|X_2|\psi_{i,\mathrm{in}}\rangle, \ldots, \langle\psi_{i,\mathrm{in}}|X_n|\psi_{i,\mathrm{in}}\rangle),$$

where n denotes the number of qubits. Then we calculate the similarity between the high-dimensional data points as we explained in Sec. II A. The low-dimensional data $\{y_i\}$ are defined similarly by the following equation using a certain constant value a:

$$y_i = a(\langle\psi_{i,\mathrm{out}}|X_2|\psi_{i,\mathrm{out}}\rangle, \langle\psi_{i,\mathrm{out}}|X_3|\psi_{i,\mathrm{out}}\rangle). \tag{15}$$

The constant value a is a hyperparameter to adjust the scale of low-dimensional data. We set the hyperparameter $a = 1$ and circuit depth $L = 8$. In this numerical experiment, we visualize the quantum states for every 1000 Trotter steps. The result is shown in Fig. 4. In this figure, we can see that two clusters corresponding to the sign of the coupling constants are formed.

2. *Visualization with infidelity metric*

The second method is to calculate the similarity of the data in high-dimensional space via infidelity of two different quantum states. The similarity between low-dimensional data is calculated in the same way as the previous case, except for the setting of hyperparameter $a = 10$. Figure 5 shows the process of the dynamics for every 1000 Trotter steps. This

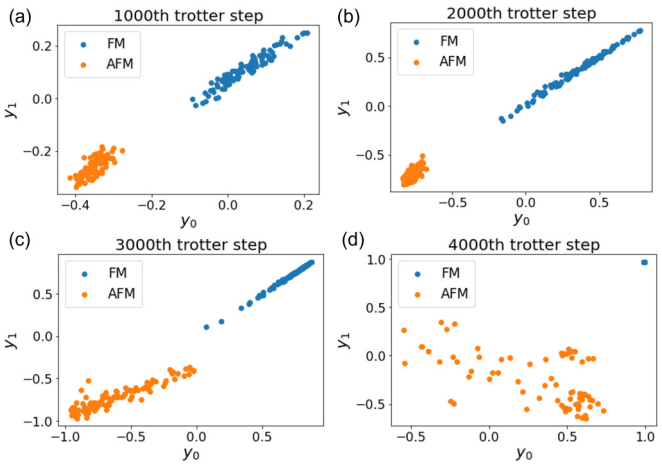

FIG. 4. Process of a time-dependent transverse-field two-body Ising model when the calculation of the similarity in high-dimensional data is based on the observables. Visualization of the quantum states of the (a) 1000th ($t = 10$), (b) 2000th ($t = 20$), (c) 3000th ($t = 30$), and (d) 4000th ($t = 40$) Trotter steps by making models at each step, respectively. The orange or blue points correspond to the cases where all J_{ij} in the Hamiltonian are positive [antiferromagnetic (AFM)] or negative [ferromagnetic (FM)].

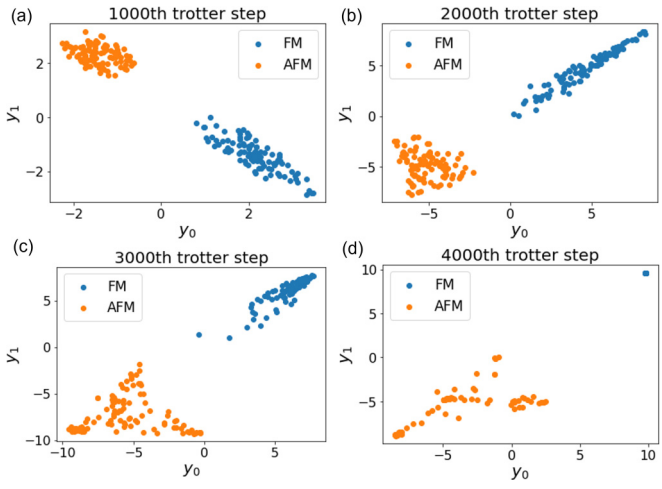

FIG. 5. Process of a time-dependent transverse-field two-body Ising model when the calculation of the similarity in high-dimensional data is based on the fidelity. Visualization of the quantum states of the (a) 1000th, (b) 2000th, (c) 3000th, and (d) 4000th Trotter step, respectively. The orange and blue points correspond to the cases where J_{ij} is positive or negative, respectively.

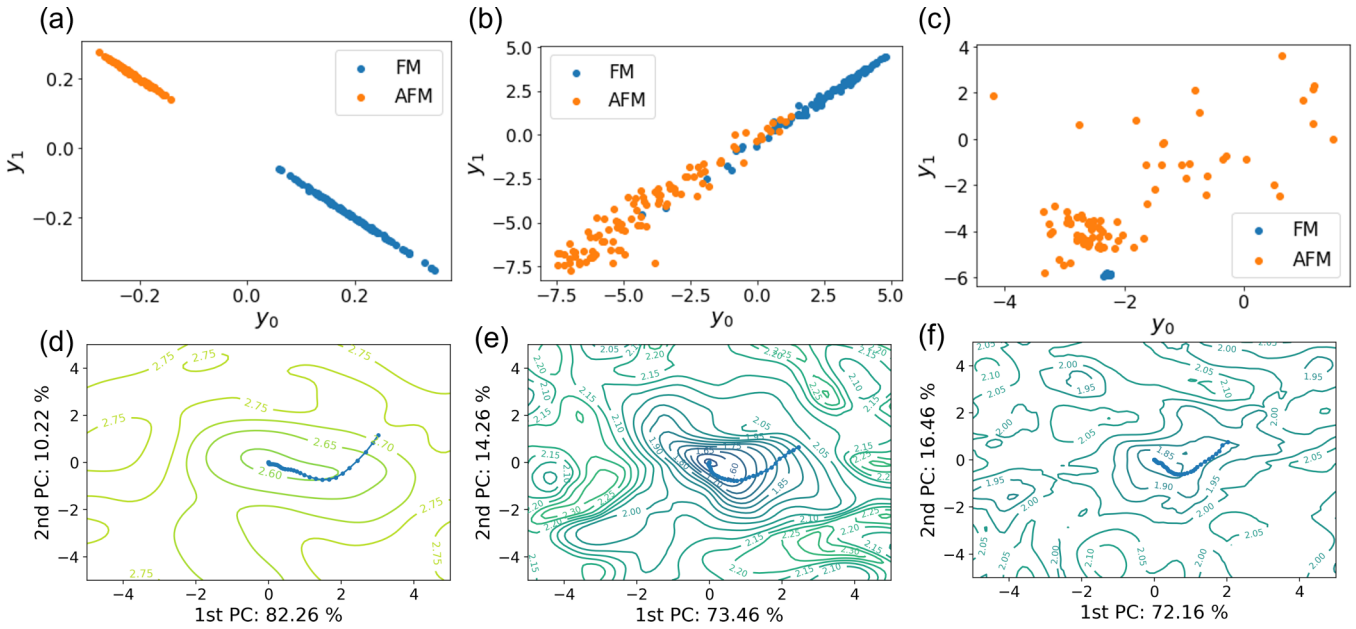

FIG. 6. Examples of undesirable visualizations. In (a), although the plot is two-dimensional, it is plotted in a one-dimensional manner. In (b) and (c), the different classes cannot be identified unless they are plotted in different colors. (d)–(f) Loss contours and optimization trajectories of (a)–(c), respectively. In all three cases, the parameters are optimized.

figure also shows the two clusters corresponding to the sign of the coupling constants.

3. *The effect of multiplying a constant value*

When we visualize the quantum data, we sometimes get undesirable visualization figures, which can be avoided by appropriately tuning the hyperparameter a. For example, we plot three examples of wrong choices of a in Fig. 6. In Fig. 6(a), the data are arranged in an almost straight line, despite the two-dimensional plot. In Figs. 6(b) and 6(c), it is impossible to determine that these data belong to different classes without the colors in the plots since each class is colored to make the figures easier to understand. One possible reason is that the cost function optimization is not sufficient to converge to a local minimum.

We examine whether the optimization converges near a local minimum. To this end, we visualize the loss contours and optimization trajectories in a two-dimensional plane. In Ref. [23], the authors developed a method to visualize loss landscape and optimization trajectories with the loss contours. They used the visualization method to investigate the relationship between loss landscape and trainability or generalization for neural networks. Before we describe the detailed method, let us define θ_i as a vector of trainable parameters at the ith epoch. We consider the matrix M consisting of $(\theta_i - \theta_n)$ for all i, where the n is the last epoch. Specifically, M is written by

$$M = [\theta_0 - \theta_n; \theta_1 - \theta_n; \ldots; \theta_{n-1} - \theta_n]. \tag{16}$$

The matrix M is transformed by principal component analysis [24], and the first and second principal component vectors are used as the axis of the visualization. Along the two principal component vectors, the loss contours and optimization trajectories are plotted on a two-dimensional plane. We use this visualization method to tune the hyperparameter a and to confirm our optimization works appropriately.

We show the visualization of the loss landscapes in Figs. 6(d)–6(f). From these figures, the optimization is sufficiently done to reach the local minima. Despite the cost function being well optimized, we have not been able to reflect enough features on the low-dimensional data to discriminate different clusters. This is attributed to the wrong design of the cost function, where the similarity is not appropriately defined. For example, in Fig. 6(a), the parameter a for adjusting the scale is multiplied by 1 instead of by 10 in Fig. 5(a). In Figs. 6(b) and 6(c), both high-dimensional and low-dimensional data are multiplied by 10 in similarity calculation, instead of by 1 in Figs. 4(c) and 4(d). These results suggest that, when calculating similarity, it is necessary to adjust the scale of the high- and low-dimensional data so that the cost function adequately reflects the characteristics of the given data.

To be more concrete, the need to adjust the scale of the high- and low-dimensional data similarities may occur in larger systems since those similarities may not reflect surrounding points. First, we refer to the high-dimensional data similarities. Since high-dimensional data similarities are likely to become smaller in simulating a larger quantum system, it is more difficult to reflect the features of higher-dimensional data into low-dimensional data. We can alleviate this difficulty by taking an n-square root of the fidelities to include more information about the surrounding points. This change affects the cost function, and the high-dimensional data similarities are easily reflected in the low-dimensional data. Next, we refer to the low-dimensional data similarities. If the low-dimensional data move a small region, optimizing the cost function is difficult because similarities between dissimilar points cannot have small values. We deal with this difficulty by multiplying observables by a constant value. For

example, let us consider Student t distribution without the normalization factor as $u = 1/(1 + v^2)$. If v ranges from -1 to 1, the range of u is $[0.5, 1]$. In this case, optimization of cost function may be difficult because the lower bound causes dissimilar points not to have small low-dimensional data similarities. We deal with this difficulty by multiplying observables by a constant value. If we multiply v by 10, the range of u becomes $[0.0099, 1]$. The lower bound of low-dimensional data similarities becomes smaller by multiplying a constant value. As a result, dissimilar points can take lower values of low-dimensional data similarities. In this way, the smaller lower bound of low-dimensional data similarities facilitates distinguishing dissimilar points in optimizing the cost function.

V. CONCLUSIONS AND FUTURE WORKS

In this paper, we have visualized classical and quantum data by a QNN. Specifically, we employed a parameterized quantum circuit as a quantum model to generate low-dimensional data, which is trained so that the similarity of the high-dimensional data is maintained. In the quantum case, the similarity of the quantum states is calculated in two ways: measuring multiple observables to obtain high-dimensional classical data from quantum states and calculating distance of two quantum states directly from fidelity.

We performed numerical simulations of two-dimensional visualizations of the Iris dataset as the case of the classical inputs and quantum states evolved under Hamiltonian dynamics as the case of the quantum inputs. It was found that the proposed method worked well for both classical and quantum inputs, with appropriate low-dimensional visualization. Specifically, in the case of quantum data, it is difficult to visualize the data well unless the constant factor is multiplied by the similarity of the low-dimensional data. This is probably because the similarity between the quantum states takes too small a value. While we treated this constant factor as a hyperparameter, it can be trainable parameters as a_i for each low-dimensional data y_i for further improvement. At least, by doing so, the performance of nonparametric t-SNE can be guaranteed in the proposed model with a two-qubit trivial circuit. This implies that the proposed method is expected to work on a relatively small quantum device. For more complex data visualization, we can improve the representation capability of quantum circuits.

One of the other possibilities using our proposed method is to compress quantum data by defining the low-dimensional data by quantum states and the similarity by a fidelity-based metric. While real quantum data, such as a set of outputs from a quantum algorithm, live in a large Hilbert space, such quantum data can be mapped into a smaller Hilbert space with a fewer number of qubits, keeping their similarity. Then quantum machine learning algorithms can be further applied to such a compressed quantum dataset. Further investigation in this direction is an intriguing future issue. The other future task is to construct a model including a decoder, which decodes data from compressed data, such as Variational AutoEncoder [25] in classical machine learning. It may be possible to create a quantum state with certain desired properties from the classical data in the middle layer.

ACKNOWLEDGMENTS

K.M. is supported by JST PRESTO Grant No. JPMJPR2019 and JSPS KAKENHI Grant No. 20K22330. K.F. is supported by JST Exploratory Research for Advanced Technology Grant No. JPMJER1601 and JST Core Research for Evolutional Science and Technology Grant No. JPMJCR1673. This paper is supported by Ministry of Education, Culture, Sports, Science and Technology Quantum Leap Flagship Program (MEXTQLEAP) Grant No. JPMXS0118067394 and JPMXS0120319794. We also acknowledge support from JSTCOI-NEXT program.

[1] L. van der Maaten and G. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. **9**, 2579 (2008).

[2] K. Ch'ng, N. Vazquez, and E. Khatami, Unsupervised machine learning account of magnetic transitions in the Hubbard model, Phys. Rev. E **97**, 013306 (2018).

[3] Y. Yang, Z.-Z. Sun, S.-J. Ran, and G. Su, Visualizing quantum phases and identifying quantum phase transitions by nonlinear dimensional reduction, Phys. Rev. B **103**, 075106 (2021).

[4] K. N. Okada, K. Osaki, K. Mitarai, and K. Fujii, Identification of topological phases using classically-optimized variational quantum eigensolver, arXiv:2202.02909.

[5] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, Nat. Phys. **17**, 1013 (2021).

[6] L. Van Der Maaten, Learning a parametric embedding by preserving local structure, in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics* (PMLR, Clearwater Beach, 2009), pp. 384–391.

[7] D. Dua and C. Graff, UCI Machine Learning Repository (University of California, Irvine, 2017), http://archive.ics.uci.edu/ml.

[8] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nat. Commun. **12**, 1791 (2021).

[9] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, Quantum embeddings for machine learning, arXiv:2001.03622.

[10] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, Data re-uploading for a universal quantum classifier, Quantum **4**, 226 (2020).

[11] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf, Quantum Fingerprinting, Phys. Rev. Lett. **87**, 167902 (2001).

[12] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell *et al.*, Quantum supremacy using a programmable superconducting processor, Nature (London) **574**, 505 (2019).

[13] S. Aaronson and A. Arkhipov, The computational complexity of linear optics, arXiv:1011.3245.

[14] M. J. Bremner, R. Jozsa, and D. J. Shepherd, Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy, Proc. R. Soc. A **467**, 459 (2011).

[15] Y. Suzuki, Y. Kawase, Y. Masumura, Y. Hiraga, M. Nakadai, J. Chen, K. M. Nakanishi, K. Mitarai, R. Imai, S. Tamiya *et al.*, QULACS: A fast and versatile quantum circuit simulator for research purpose, Quantum **5**, 559 (2021).

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, PYTORCH: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems 32*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., Red Hook, 2019), pp. 8024–8035.

[17] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, Sharpness-aware minimization for efficiently improving generalization, arXiv:2010.01412.

[18] J. Kwon, J. Kim, H. Park, and I. K. Choi, ASAM: Adaptive Sharpness-Aware Minimization for scale-invariant learning of deep neural networks, Proc. Mach. Learn. Res. **139**, 5905 (2021); https://github.com/davda54/sam.

[19] D. P. Kingma and J. Ba, ADAM: A method for stochastic optimization, arXiv:1412.6980.

[20] T. Kadowaki and H. Nishimori, Quantum annealing in the transverse Ising model, Phys. Rev. E **58**, 5355 (1998).

[21] H. F. Trotter, On the product of semi-groups of operators, Proc. Am. Math. Soc. **10**, 545 (1959).

[22] M. Suzuki, Generalized Trotter's formula and systematic approximants of exponential operators and inner derivations with applications to many-body problems, Commun. Math. Phys. **51**, 183 (1976).

[23] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, Visualizing the loss landscape of neural nets, arXiv:1712.09913.

[24] K. Pearson, Liii. on lines and planes of closest fit to systems of points in space, London Edinburgh Dublin Philos. Mag. J. Sci. **2**, 559 (1901).

[25] D. P. Kingma and M. Welling, Auto-encoding variational Bayes, arXiv:1312.6114.